Tampereen yliopisto
Tampere University

Welcome to

Faculty of Information Technology and Communication Sciences (ITC),

Computing Sciences Unit

Tampereen yliopisto
Tampere University

TIE-02306

Introduction to Software Engineering

2019, 1-2. periods

5 credit units

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems, IoT
13. Recap

Tampereen yliopisto
Tampere University

1. Course basics, intro

- course goals
- course requirements
- grading
- lectures
- weekly exercises
- project assignment = exercise work
- hints for group work
-
-
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. **Sw Eng in general, overview**
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

2. Sw Eng in general, overview

- software engineering vs. ”other” work…
- sw eng IS teamwork
-
- how to get sw; buy (build), tailor/modify, COTS, SaaS,…
-
- it MAY be worth thinking to modify your way of work, not the software…
-
- ethics (Code of Conduct)
-
- documentation
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. **Requirements**
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

3. Requirements (SRS = sw req. specification)

- Preliminary Study / Feasibility Analysis
- requirements elicitation
- stakeholders and stakeholder analysis
- Requirements
  - functional
  - non-functional
  - restrictions
- process
- safety vs. security
- usability
- producer's responsibilities
- customer's responsibilities
- requirements management... traceability... jäljitettävyys ??

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. **Different software systems**
5. Basic UML Diagrams ("Class", Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

4. Different kind of software systems

- Different kind of software systems
  - stand-alone
  - client-server
  - mobile
  - distributed
  -
- mission-critical / safety-critical systems
-
- architecture
-
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. **Basic UML Diagrams (”Class”, Use Case, Navigation)**
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

5. Basic UML Diagrams

- ”entity diagram”, class diagram
- use case diagram and User Stories
- navigation diagram
-
-
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. **Life Cycle models**
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

6. Life Cycle models (etc.)

- software life-cycle
- software development life-cycle (SDLC)
- different models
  - waterfall
  - spiral
  - iterative
  - incremental
  - agile (e.g. Scrum) Scrum in more detail
  - lean
  - DevOps
  - SAFe
  - kanban
  - hybrid models…
- maintenance

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. **UML diagrams, in more detail**
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

7. UML diagrams, in more detail

- Just a few more UML diagrams…
  - state diagrams
  - scenarios
  - sequence diagrams
  - activity diagrams
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. **Quality and Testing**
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

8. Quality and Testing

- Quality Assurance (QA)
- testing
  - review / inspection
  - unit testing
  - integration testing
  - system testing
  - usability testing
  - acceptance testing
- TDD (Test-Driven Development)
- version control
- configuration management
- verification and validation (V&V)
-
- standards (use as a checklist)
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. **Project work**
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

9. Project work

- working in a project
- from group work to teamwork
- common problems / pitfalls
-
- soft skills = people (”Human matters”)
-
- patterns
-
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. **Project management**
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

10. Project management

- overview of a project
- common problems (How to Fail)
- common success factors
-
-
- scheduling / time estimations
- change management
- metrics / measurements (complexity, FPA, FiSMA, COCOMO,...)
- risk management
-
- standards
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. **Open source, APIs, IPR**
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

11. Open source, APIs, IPR

- Open Source (OS),
- Open Data, Big Data
- API = application programming interface
-
-
- IPR = intellectual property rights
-
- components
- reuse
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. **Embedded systems, IoT**
13. Recap

Tampereen yliopisto
Tampere University

12. Embedded systems, IoT

- sw + hw
- embedded systems
- real-time systems (RT)
-
- IoT = internet of things / IoE = internet of everything
-
- AI (artifical intelligence)
-
-

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems
13. **Recap**

Tampereen yliopisto
Tampere University

13. Recap, summary, resume

What you should remember from this course…

- life cycle phases in general (analysis, requirements, design, implementation, testing, deployment, maintenance), and supporting tasks: hw, documentation, training
- some SDLC models
-
- three classes of requirements; functional, non-functional, restrictions
-
- common problems in sw dev
- common success factors in sw dev
-
- customer also has some responsibilities !
-

Tampereen yliopisto
Tampere University

< course starts at next slide >

Tampereen yliopisto
Tampere University

Welcome to

Faculty of Information Technology and Communication Sciences (ITC),

Computing Sciences Unit,

Hervanta campus

Tampereen yliopisto
Tampere University

TIE-02306 ”ItSE”

Introduction to Software Engineering

2019, 1-2. periods

5 credit units

01-intro-ItSE-2019-v7

Tampereen yliopisto
Tampere University

Welcome to

Faculty of Information Technology and Communication Sciences (ITC),

Computing Sciences Unit,

Hervanta campus

Tampereen yliopisto
Tampere University

Course contents (plan)

1. **Course basics, intro**
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. Project management
11. Open source, APIs, IPR
12. Embedded systems, IoT
13. Recap, summary

2nd lecture time is explaining project assignment

Tampereen yliopisto
Tampere University

1. Course basics, intro

- course goals
- course requirements
- course personnel 2019
- grading
- lectures
- weekly exercises
- project assignment = exercise work
- hints for group work
- what to do next ?
-

Tampereen yliopisto
Tampere University

TIE-02306 ”ItSE” course goals

This is the basic course to Software Engineering (Sw Eng), suitable also for minor study subject readers.

Learning outcomes of the course: After the course student understands how different types of software is developed in a professional way and what kind of knowledge and skills it requires. The ultimate goal is efficient participation in software projects in various stakeholder roles, for instance as an expert or customer. We want you to understand the whole SDLC and its processes, which is much more than just coding in the dark basement.

“From internet you can get information, but knowledge and wisdom you get from lectures.”

Everybody should have an understanding about software development life cycle and processes. TIE-02306 gives you that view, as well as a more detailed touch to requirements, which are needed and understood by all stakeholders in every sw eng project.

Tampereen yliopisto
Tampere University

So you want to be an engineer…

Find at YouTube ”Dilbert knack the engineer”

…who also understands software processes (software business you will learn at other faculties)**.**

Tampereen yliopisto
Tampere University

Software engineers are needed in future !

POPULAR SCIENCE SUBSCRIBE

CARS

SOFTWARE NOW TO BLAME FOR 15 PERCENT OF CAR RECALLS

YOU CAN'T JUST HOLD THE HOME AND LOCK BUTTONS TO SOLVE THIS ONE

Bengt Halvorson / The Car Connection June 2, 2016

Tesla driver dies in first fatal crash while using autopilot mode

The autopilot sensors on the Model S failed to distinguish a white tractor-trailer crossing the highway against a bright sky

ARIZONA · 1 day ago

Self-driving Uber car kills Arizona pedestrian, police say

TRANSPORTATION \ UBER \ RIDE-SHARING

Uber's self-driving car showed no signs of slowing before fatal crash, police say

The vehicle was traveling at 40 mph

By Andrew J. Hawkins | @andyjayhawk | Mar 19, 2018, 7:09pm EDT

OPINION

Uber self-driving car accident: Who's to blame when there's no driver?

The Conversation By Raja Jurdak and Salil S. Kanhere

Posted Tue at 7:55am

Tampereen yliopisto
Tampere University

Software engineers are needed in future !

WIRED

What Boeing's 737 MAX Has to Do With Cars: Software

AARIAN MARSHALL TRANSPORTATION 06.30.19 07:00 AM

WHAT BOEING'S 737 MAX HAS TO DO WITH CARS: SOFTWARE

THE WALL STREET JOURNAL.

Home World U.S. Politics Economy Business Tech Markets Opinion Life & Arts Real Estate

BUSINESS

FAA Finds New Software Problem in Boeing's 737 MAX

Plane maker agrees to address the problem and believes it can be fixed with a software tweak

By Andrew Tangel and Andy Pasztor

Updated June 26, 2019 9:55 pm ET

Boeing Co. and federal regulators said they have identified a new software problem on the 737 MAX, further delaying the process of returning the troubled jet to service.

BUSINESS NEWS JUNE 27, 2019 / 5:30 PM / 2 MONTHS AGO

Boeing sees fix for latest 737 MAX software flaw in September

David Shepardson, Eric M. Johnson, Tracy Rucinski 5 MIN READ

(Reuters) - Boeing Co (BA.N) will take until at least September to fix a newly identified problem on its grounded 737 MAX, a company official told Reuters, meaning the workhorse jet's return to service will be delayed until October at the earliest, significantly longer than most airlines had expected.

Bloomberg

Technology

Boeing's 737 Max Software Outsourced to $9-an-Hour Engineers

By Peter Robison

June 28, 2019, 11:46 PM GMT+3

CNN politics 45 Congress Supreme Court Facts First 2020 Election

New flaw discovered on Boeing 737 Max, sources say

By Shimon Prokupecz, Drew Griffin and Gregory Wallace, CNN

Updated 0719 GMT (1519 HKT) June 27, 2019

Tampereen yliopisto
Tampere University

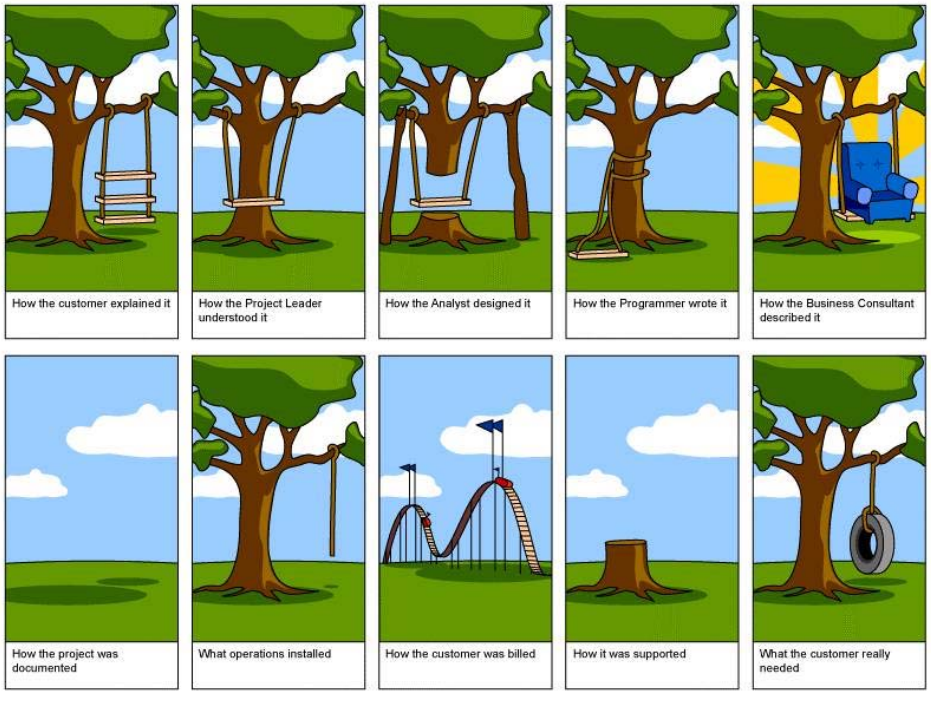
Main point: requirements

Why so many software projects fail ?

In many cases it is about **requirements and the process**.

And in many such cases, it is also the **customer's** fault, too. Not committed.

An active and knowledgeable customer is needed for success !



Tampereen yliopisto
Tampere University

developer vs. customer about the product

Well, good requirements are for example

- complete
- accurate
- unambiguous
- clear
- easy to understand
- short
- and much more…

That is easier to say than do.

Hint: don't argue, negotiate.

Tampereen yliopisto
Tampere University

Course structure and requirements

- **Lectures** (volunteer)
  - slides to Moodle
  - lecture recordings via ECHO360
- **Weekly exercises** (WE, volunteer)
  - supports lectures and exercise work
  - no homework, stress yourself only in WE times ;-)
  - presence logging by Moodle tool (QR)
- **Project assignment** = exercise work (practical work)
  - made in groups of four (4) students
  - requirements specification document
  - UML diagrams: Concept (Entity), Navigation, Use Case
  - **tools: Trello, Dia (/Enterprise Architect), PRP,…**
- **Exam** (electronic EXAM or Moodle exam), 2..3 exams during the course.

Tampereen yliopisto
Tampere University

Course structure and requirements, 1

- **Lectures** (volunteer)
  - Pdf slides to Moodle
  - lecture recordings via ECHO360.

All course information (primary source) is in TUNI Moodle;

moodle.tuni.fi/

Tampereen yliopisto
Tampere University

Course structure and requirements, 2

- **Weekly exercises** (WE, volunteer)
  - supports lectures and exercise work
  - no homework, stress yourself only in WE times
  - most of work is made is small groups
  - presence logging by Moodle tool (QR-code).

If you plan to attend weekly exercises, apply/sign for some exercise class. Signing at Moodle opens after first lecture (27.08.2019 at 1800 o'clock). WE signing at Moodle ends on Thursday 29.08. at 2355 o'clock. Depending of the amount of students at this course and sign-ups for WE groups/classes, we are not going to establish all five WE groups. We will inform students at Moodle on Friday 30.08. which WE groups exist (WE1, week 36).

Responsible assistant: Ulla.

Tampereen yliopisto
Tampere University

Kahoot! Quiz. *Will come soon…*

Are you going to attend Weekly Exercises ?

How many years you have studied at university ? 0 this is first, 1, 2, 3or+

Is Software Engineering: Science, Art, Folklore ?

Have you used tools; Trello, Dia, PRP earlier ?

Tampereen yliopisto
Tampere University

Course structure and requirements, 3

- **Project assignment** = exercise work (practical work)
  - made in groups of four (4) students
  - requirements documentation (not exactly full specification)
  - made in two phases
  - first presentation and final (2.) presentation
  - UML diagrams: Use Case, Concept (Entity), Navigation
  - document delivery in pdf format at TUNI Moodle
  - presentation slides' delivery in pdf-format at PRP (via Moodle).

- **tools:**
  - **Trello ”project management”**
  - **Dia (/Enterprise Architect) for UML diagramming**
  - **PRP (Peer Review Program) peer feedback, self assesment.**

- **Responsible assistant: Ulla.**

Tampereen yliopisto
Tampere University

Vote deadline for project assignment

Vote at Moodle;

When (**weekday and time**) would be the **deadline** for project assignment (exercise work) deliveries ?

- Friday at 1600 o'clock
- Friday at 2355 o'clock
- Saturday at 2355 o'clock
- Sunday at 2355 o'clock.

Voting is open: 20.08. from 0700 to 31.08. to 2355 o'clock.

Tampereen yliopisto
Tampere University

Course structure and requirements, 4

- **Exam** (electronic EXAM or Moodle exam)
  - Three (3) exams during the course, no "one big bang" at the end.
  - one exam will be about diagrams (Dia tool should be at EXAM classes).

1. Sw Eng basics and requirements. (weeks 41-43 ??)
2. Diagrams (Dia tool at EXAM classes… hopefully) (weeks 44-46 ??)
3. Other relevant Sw Eng matters. (weeks 47-49 ??)

2nd and 3rd possibilities (to repeat exam) would be one larger exam covering the whole course, at some times during Spring 2020.

Tampereen yliopisto
Tampere University

Grading

Student collects points from

- project assignment
- exam(s)
- weekly exercises.

From exam a certain minimum amount of point is required to pass.

Total grade (fail, 1-5) comes by amount of points gathered along the course.

Tampereen yliopisto
Tampere University

Points for grading the ItSE course 2019

- project assignment; max. 13, minimum requirement at least 5 points ???
- exams; max. 18 (6+6+6), minimum at least 2 points from every exam
- weekly exercises; max. 5, totalling 31/36 points at maximum.

Grading: ???

- 1: 15-
- 2: 20-
- 3: 24-
- 4: 27-
- 5: 30-

Points and scaling at the moment, 27 Aug 2019.

Tampereen yliopisto
Tampere University

In general…

If you don't understand something (e.g. at Moodle page or project assignment guidelines), **ask** !

At corridors all kind of rumours and "war-stories" exist about university courses, heard from students who have heard those from some "reliable source". Oh boy. ;-)

It is much better to ask before acting or neglecting something.

Tampereen yliopisto
Tampere University

Kahoot! Quiz.

Are you going to attend Weekly Exercises ?

How many years you have studied at university ? 0 this is first, 1, 2, 3or+

Is Software Engineering: Science, Art, Folklore ?

Have you used tools; Trello, Dia, PRP earlier ?

Tampereen yliopisto
Tampere University

A project with unclear requirements is like …

A project with unclear requirements is like a driving with poor visibility of your destination.

[Galileosystems]

Tampereen yliopisto
Tampere University

History of the course

1990- 81210 ”OTuPK”

... TIE-02300 ...TIE-02301 ... ”JOTU”

2019- TIE-02306 Introduction to Software Engineering ”ItSE”

The first English implementation.

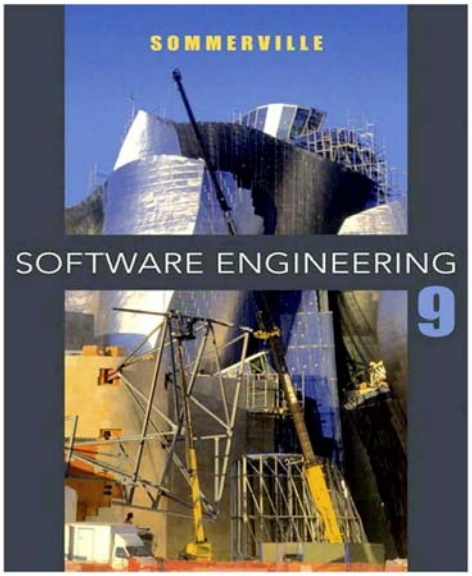
At the Finnish JOTU courses there have been some 250 students annually; about 100 ICT students and 150 from other faculties. So later at your work many of you will be in some ICT project as a customer, and not as a software professional (in some vendor/developer company).

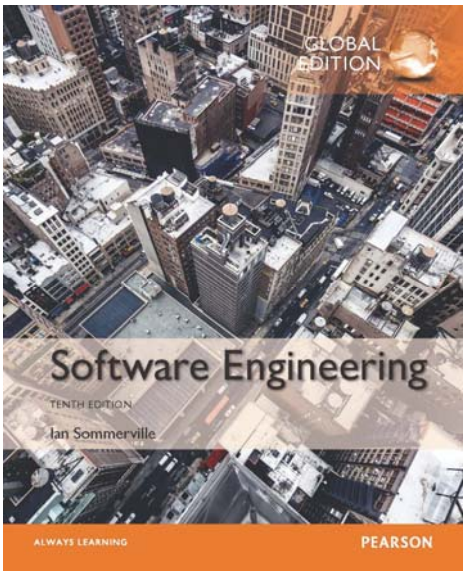
Sw Eng books

There are tens if not hundreds of ”Software Engineering” textbooks.

Some can be found from TUNI electronic library as eBooks.

Ian Sommerville’s book is the most popular one.





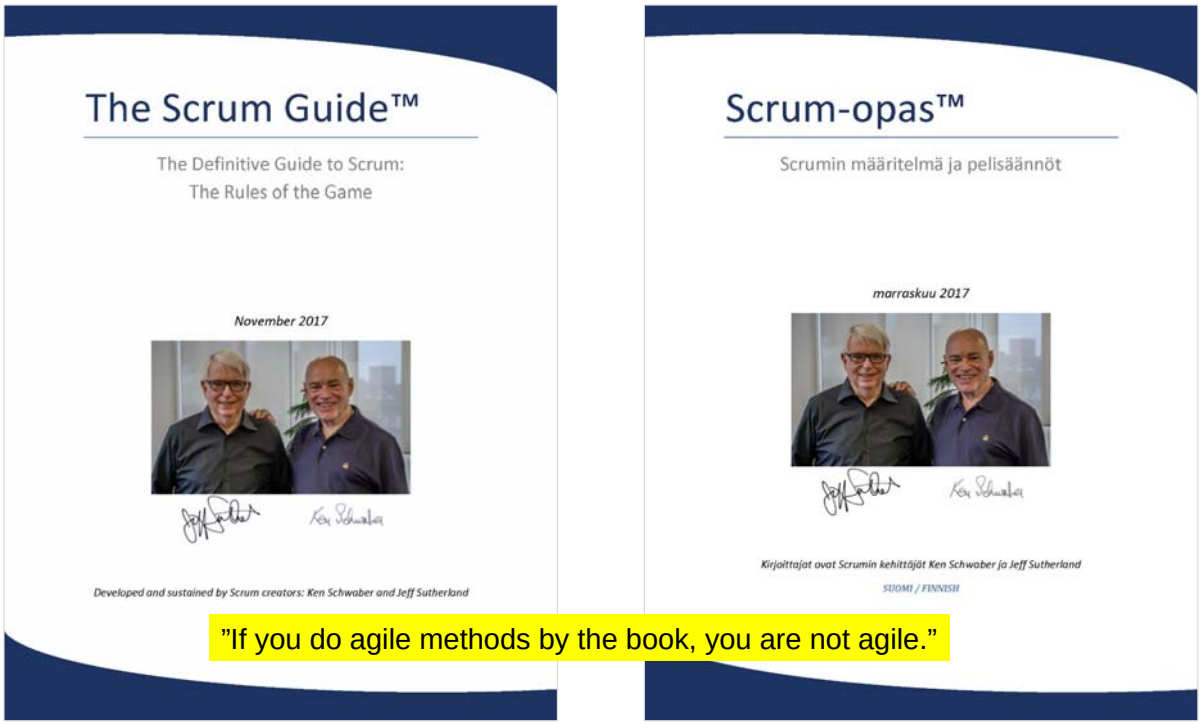
Other material

See also ”**Additional material**” section at course Moodle page. It includes a lot of miscellaneous links.

Note also **TUNI electronic library**
- eBooks (some Sw Eng books available)
- SFS Online (standards)
- IEEEXplore (IEEE publications).

Do not forget all kind of sw eng comics/cartoons, e.g.
- Dilbert
- geek-and-poke
- Glasbergen
- PhD comics.

www.scrumguides.org



”If you do agile methods by the book, you are not agile.”

Tampereen yliopisto
Tampere University

Course staff 2019

- Tero ”Tensu” Ahtee, **lectures** / **weekly exercises**
- Ulla-Talvikki Virta, **project assignment** / **weekly exercises**
- Valentina Lenarduzzi, **project assignment** / **weekly exercises.**

Course staff presents themselves at lectures at 03.09.2019.

For questions about the course, contact **Tensu**, ***tero.ahtee@tuni.fi*** .

Tensu's background, TTKK – TTY – TAU/TUNI

Studies

- 1984 - 1986 engineer (TTOL, current TAMK)
- 1987 - 05/1990 MSc (UTA, current TUNI Central campus)

Academic positions at TUT – TUNI

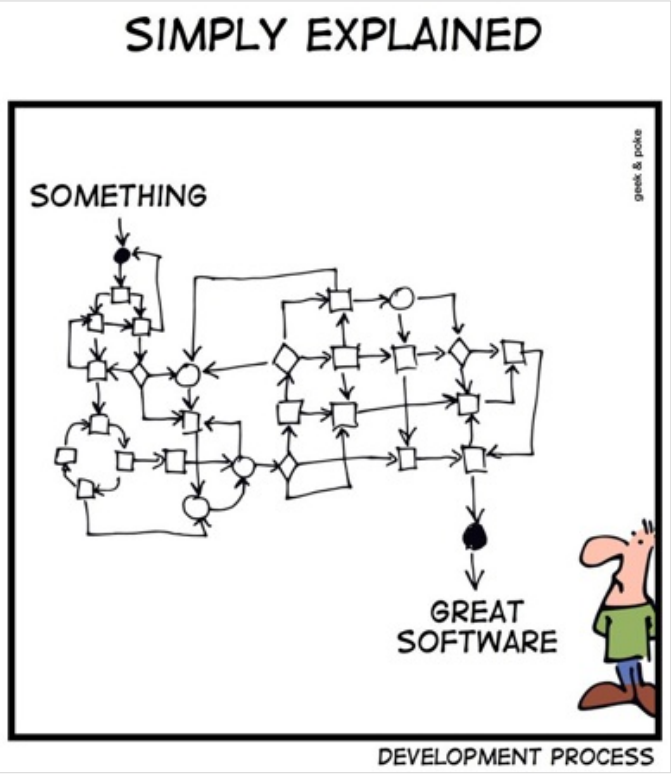
- 08/1990- acting assistant
- 1993 - assistant
- 08/1999- lecturer.

Teaching at courses

- Ohjelmistotuotannon peruskurssi 1990-
- Ohjelmistotuotannon menetelmät 1991-
- Ohjelmistotekniikan työkurssi 1990-95
- Ohjelmistotuotannon projektityö - Ohjelmistotekniikan projektityö – Tietotekniikan projektityö – Project Work on Pervasive Systems 1991-
- Projektinhallintaseminaari 2005-2013
- Demola Project Work 2008-2018.

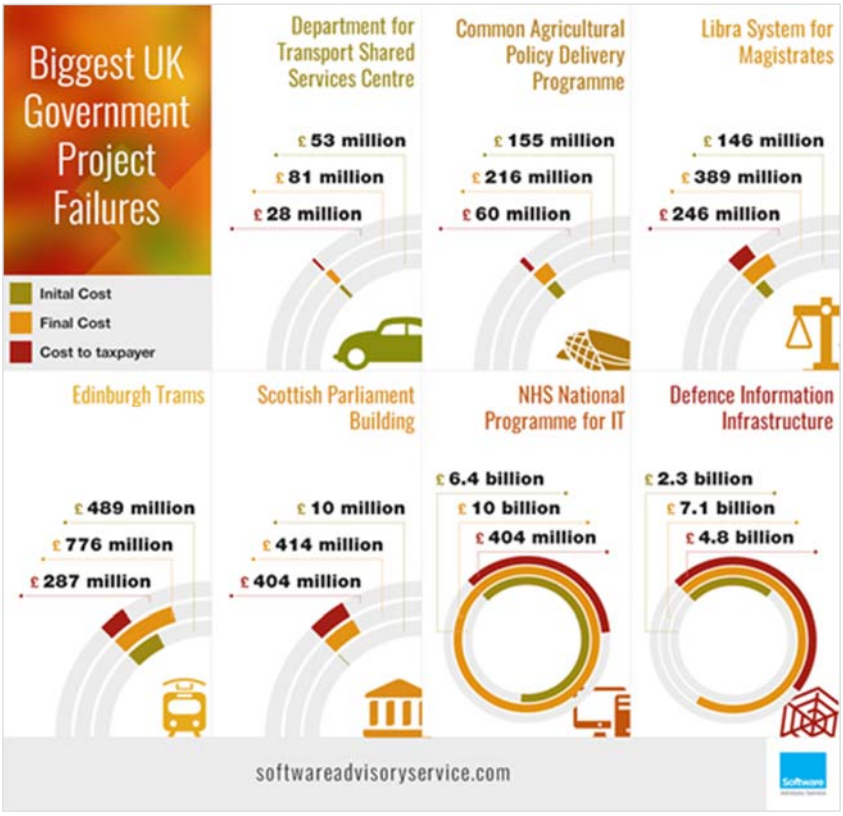
Well, of course you need a bit of good luck in software engineering projects… but the technical basics you need to know anyway.

Quite well-know method ”Desperate hacking”, in Finnish ”Läpisekoilun periaate”.



Tampereen yliopisto
Tampere University

Something you would not like to read from newspapers… and definitely not for those projects YOU work for.



SLC = software life cycle

software life cycle. The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase. *Note:* These phases may overlap or be performed iteratively. *Contrast with:* **software development cycle.**

Now you think has anything changed from 1990s ?

Well, in **every** Sw Eng project there is some
- **preliminary analysis**
- **requirements specification**
- **design**
- **implementation**
- **testing**
- **documentation**.

But methods and processes, way of working, have changed.

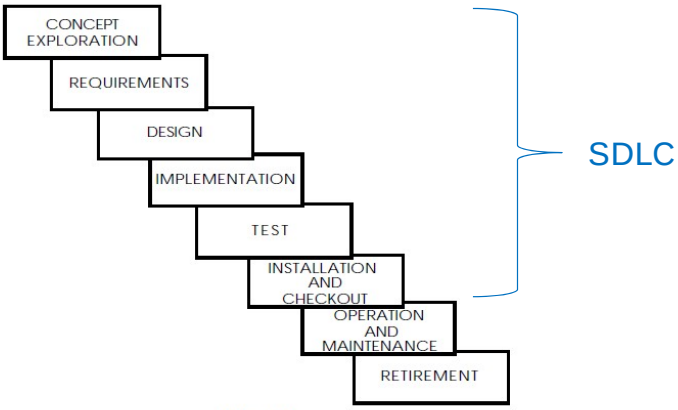


Fig 15
Sample Software Life Cycle

[**IEEE Standard Glossary of Software Engineering Terminology,** Std 610.12-1990(R2002)]

SLC = software life cycle

software life cycle. The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase. *Note:* These phases may overlap or be performed iteratively. *Contrast with:* **software development cycle.**

TIE-02306 ItSE is about that small part

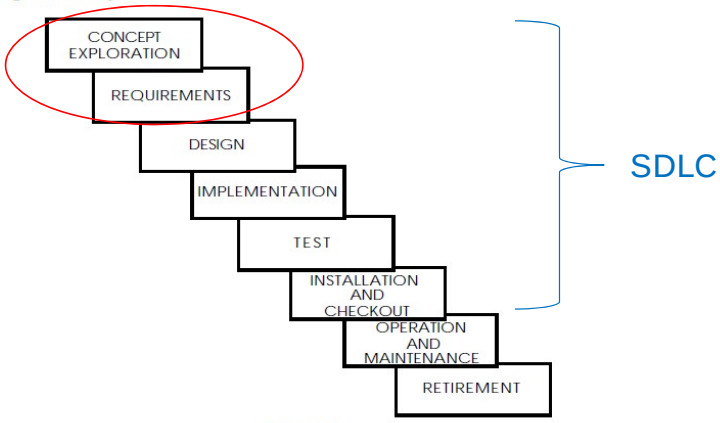


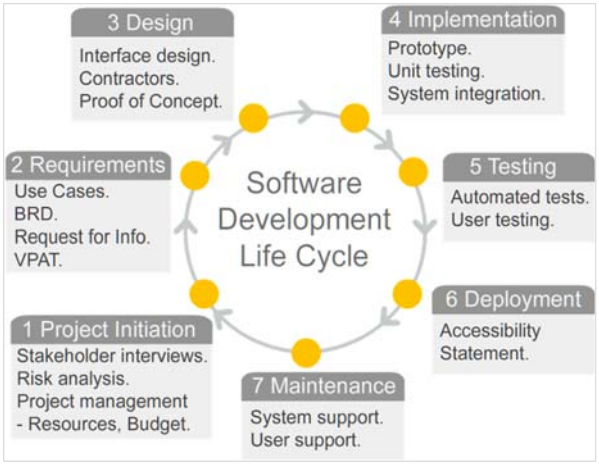
Fig 15
Sample Software Life Cycle

[**IEEE Standard Glossary of Software Engineering Terminology,** Std 610.12-1990(R2002)]

"Waterfall" shaped as a SDLC circle

Well, in **every** Sw Eng project there is some
- **preliminary analysis**
- **requirements specification**
- **design**
- **implementation**
- **testing**
- **documentation**.

[Husson University]



[University Of Melbourne]

In every life cycle model there are the same basic items in some role.

BRD= business requirements document
VPAT= voluntary product accessibility template.

Software life cycle

ISO/IEC/IEEE 24765:2017(E)
Systems and software engineering — Vocabulary
Second edition, 2017

3.3803
software development cycle
SDC = SDLC
1. period of time that begins with the decision to develop a software product and ends when the software is delivered
cf. software life cycle

3.3823
software life cycle (SLC)
SLC
1. project-specific sequence of activities that is created by mapping the activities of a standard onto a selected software life cycle model (SLCM) *[IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes,* 3.2*]*
2. software system or software product cycle initiated by a user need or a perceived customer need and terminated by discontinued use of the product or when the software is no longer available for use.

Finnish study among companies (2017)

What kind of skills are needed / required / wanted from new workers

1. Attitude, will to learn new, not giving up

2. Problem solving skills

3. Teamworking skills, communication skills.

In sw eng projects, TECHNICAL matters seldom break… it is the other matters that causes problems.

Also ”soft skills” are needed in ICT projects !

Tampereen yliopisto
Tampere University

Project Assignment = Exercise Work

Well, everything you do for the first time,
seems to require a lot of work and be difficult.
But the second time later on will be much easier. ;-)
So you have better to start early.

Tampereen yliopisto
Tampere University

Hints for group work

- contact other groupmembers quickly, get together
- find out information about the documentation and other tools; agree what to use, how those function, help-files and/or manuals
- **agree goals; e.g. grade 1 or 5**, or weekly hours effort
- work division; Scrum Master (now also Project Manager)
- a fixed (same weekday and time) **weekly meeting** is **highly recommended**
- perhaps a common calendar, for project work and times ”not available” (e.g. exams, travelling, hobbies)
- **communication tools within the group** (e.g. slack, telegram, whatsapp, SMS, e-mail,...)
- perhaps project Start Sauna Party (”kick-off”) ?

Causes of conflicts [Borg et al., 2011]

- ambition differences
- cultural differences
- bad communication
- strong wills
- unclear goals
- different prior knowledge (studies)
- aversion / dislike towards methods / tools.

Remember at group work...

If group-members have very different neeeds, it may lead to problems.

Possible solutions to conflicts [Borg et al., 2011]

- **state group rules early & clearly**
- **encourage to discuss problems**
- **well-defined roles**
- **start work with small tasks.**

”World is full of friends you have not yet met.”

You may well co-operate with fellow students, but NOT COPY other’s work.

Agile way, Scrum

The Agile idea is (see Agile Manifesto)

- ask customer what (s)he wants = customer collaboration
- make the application in iterations (”small steps”)
- after every iteration, show customer a demo, and listen comments
- change requirements if necessary = respond to change
- documentation is kept at minimum (but some have to be done).

Note that it is supposed that **customer** is active, too.

Agile way, Scrum

The Scrum idea is

- put all work items to Product Backlog (PB)
- take only few items to Spring Backlog (SB)
- do one task at a time from SB
- after every Sprint you can show a demo to customer
- take customer feedback, modify PB if necessary.

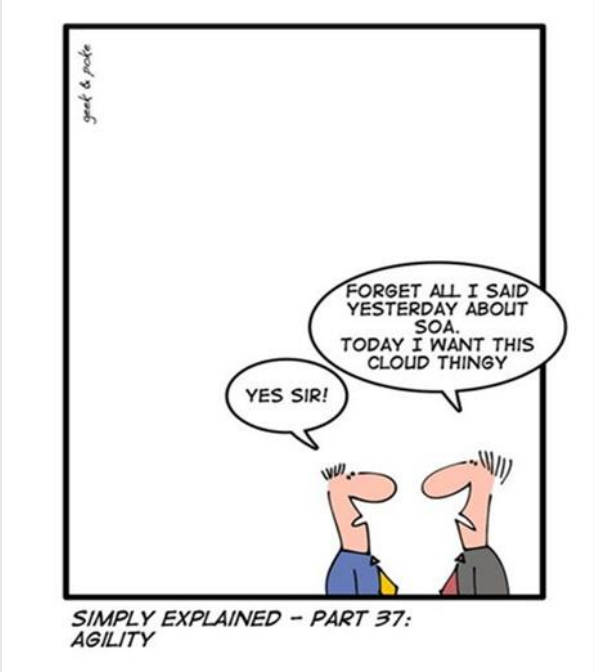
Basicly, Scrum was mentioned to experienced teams, who have already been working together on some projects, so that they know each other's strengths.

Agile way, Scrum

Agile, good and yes, but do not run into hasty conclusions.

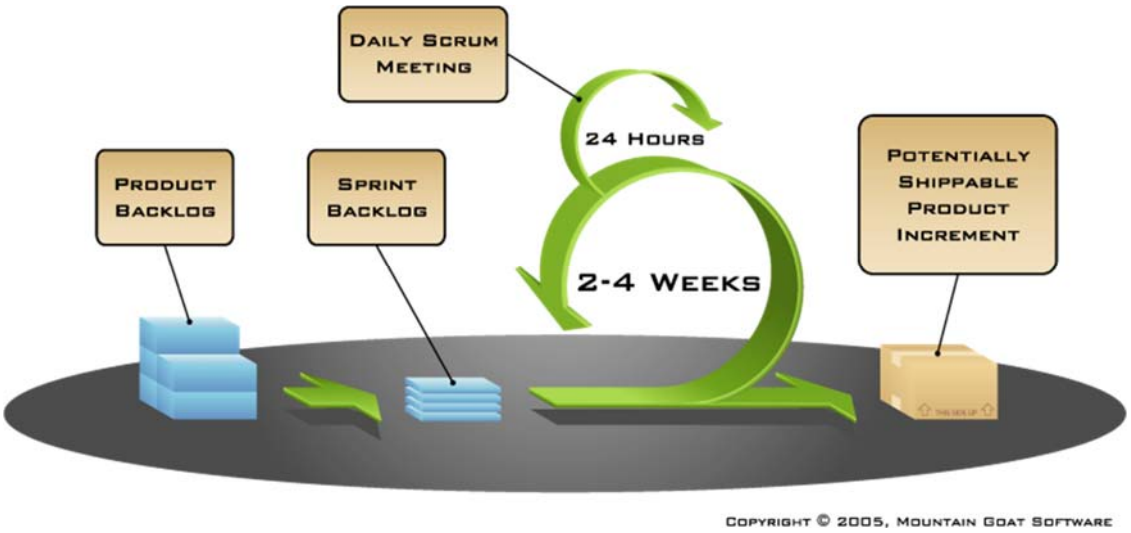
You still have to think a while before acting.

Check things first.



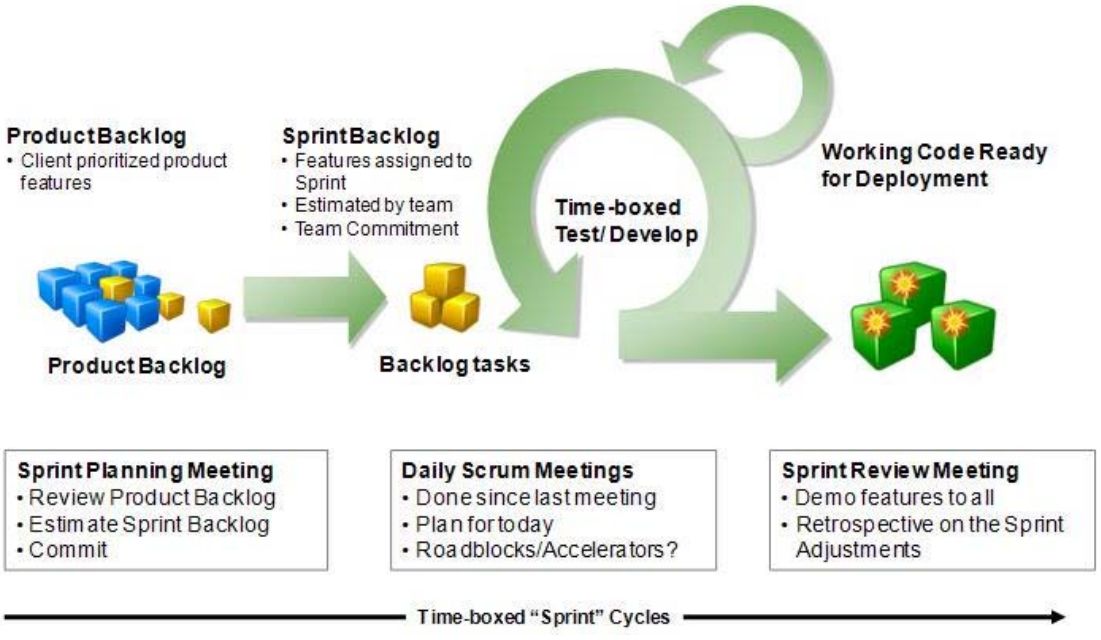
Occasionally old inventions are found and given a new "hype" name.

Putting it all together

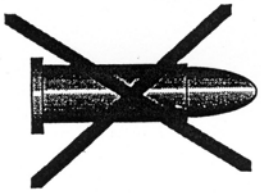


[www.mountaingoatsoftware.com/scrum]

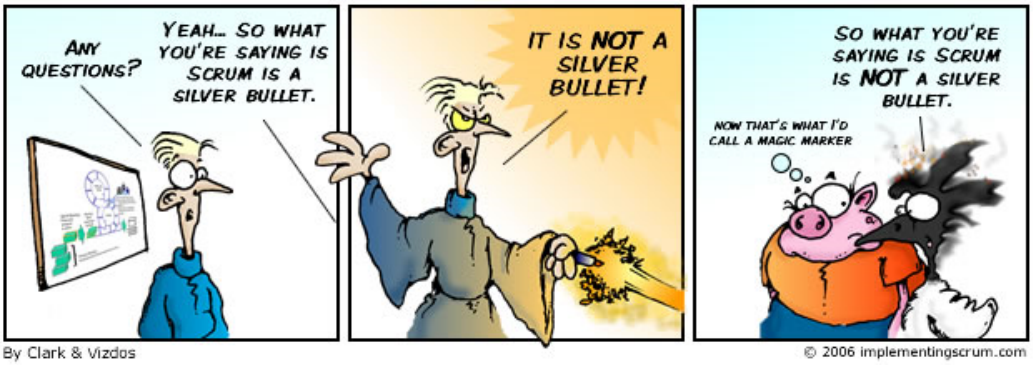
Scrum basics



There is no silver bullet!



No tool, no method will save you from having to think!



Tampereen yliopisto
Tampere University

Problems in Sw Eng projects (study 2013)

Customer point of view:

- Schedule overrun
- Budget overrun
- Different thoughts about project goals
- Lack of quality

Vendor (developer company / software company) point of view:

- Lack of communication
- Different thoughts about project goals
- Schedule overrun
- Personnel changes
- Lack of quality.

Tampereen yliopisto
Tampere University

Software Engineering -- ohjelmistotuotanto

Software (FI: ohjelmisto)

1. computer programs, procedures and possibly associated documentation and data pertaining to the operation of a computer system.

2. all or part of the programs, procedures, rules, and associated documentation of an information processing system.

3. program or set of programs used to run a computer.

Software Engineering (FI: ohjelmistotuotanto)

1. systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software.

2. application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

[IEEE 24765-2017]

Tampereen yliopisto
Tampere University

Software engineering =

- Requirements specefication
- Programming
- Quality assurance
- Documentation
- Project management
- Deployment
- Sales (and after sales)
- Maintenance.

Sw Eng is NOT just hacking/programming

Requirements specification =

Finding out needed/required features, and defining those.

Tampereen yliopisto
Tampere University

What is Software Engineering ?

Sw Eng is NOT just hacking / programming

requirements specification

Life cycle models

skillfull programming

Algorithms

Data structures

Programming languages

Architectures

teamwork

Project management

quality assurance

Testing

Validation

developer

customer

Tampereen yliopisto
Tampere University

Why making software is so hard / difficult ?

Software is abstract (compare to building a house)

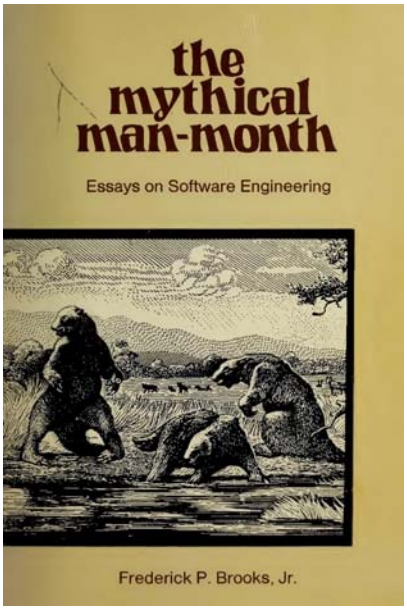
- **developers and users have diffirent views and opinions**
- **estimating effort is difficult.**

Software is dynamic

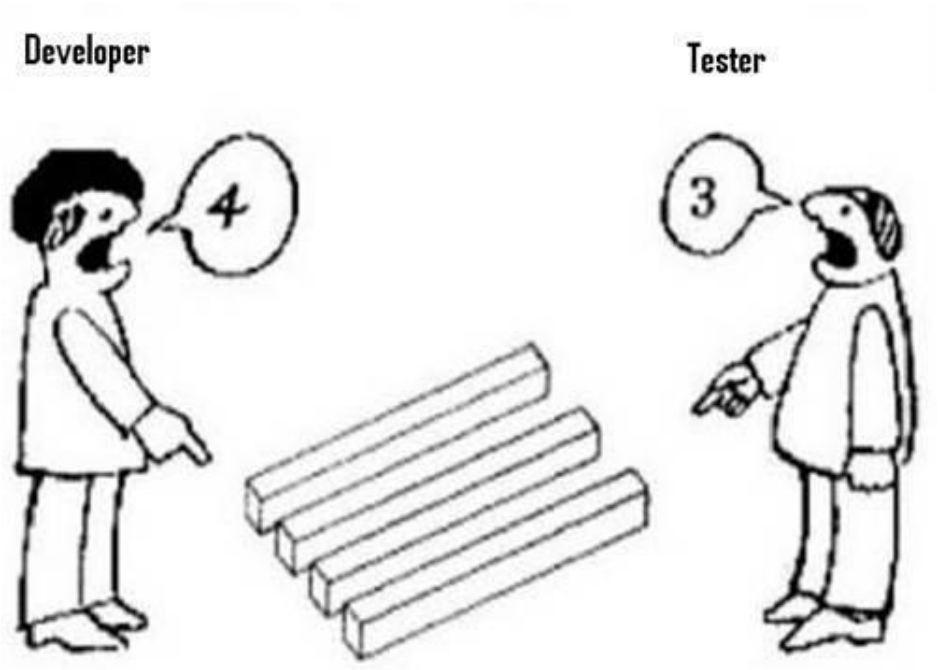
- **may be changed "easily"**
- **others suppose sw is "flexible"**

Software development is not scalable

- **adding more workers do not speed up much**
- **more workers = more communication**
- **small team tactics do not necessarily work with large sw projects.**



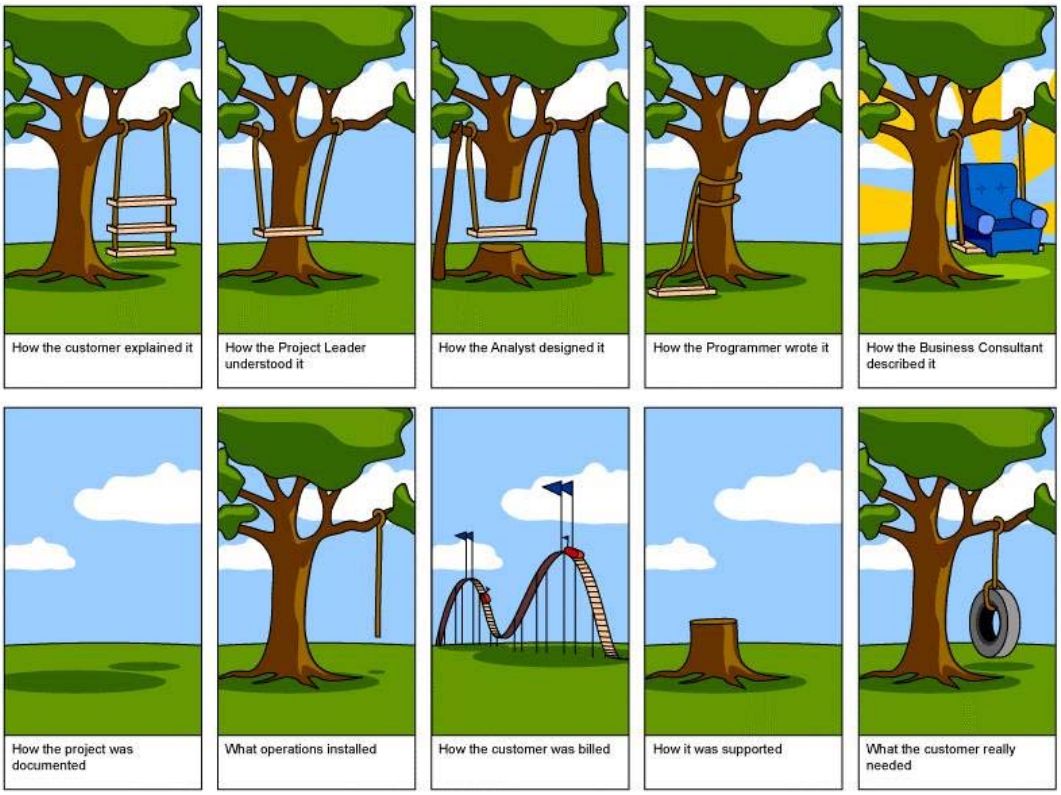
Who is right ?



Famous phrase anecdote

"To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem."

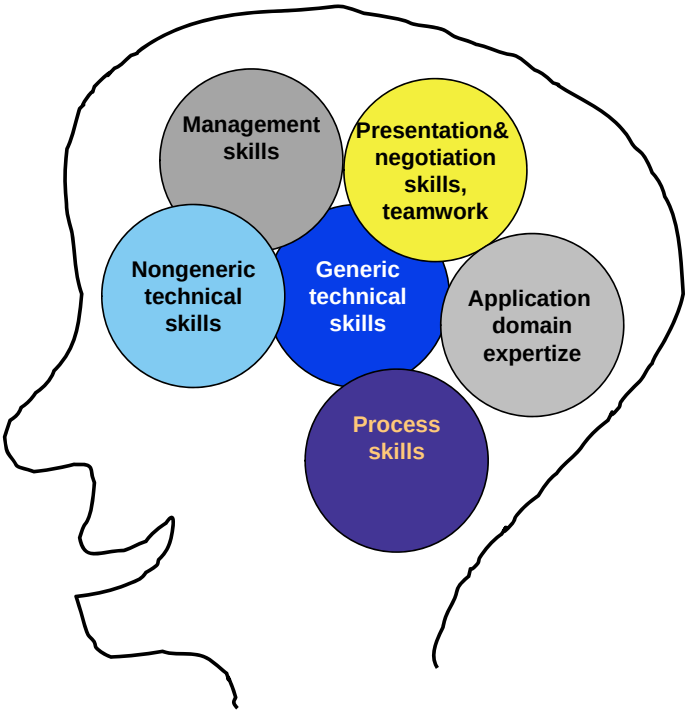
-- E. Dijkstra, 1972 Turing Award Lecture

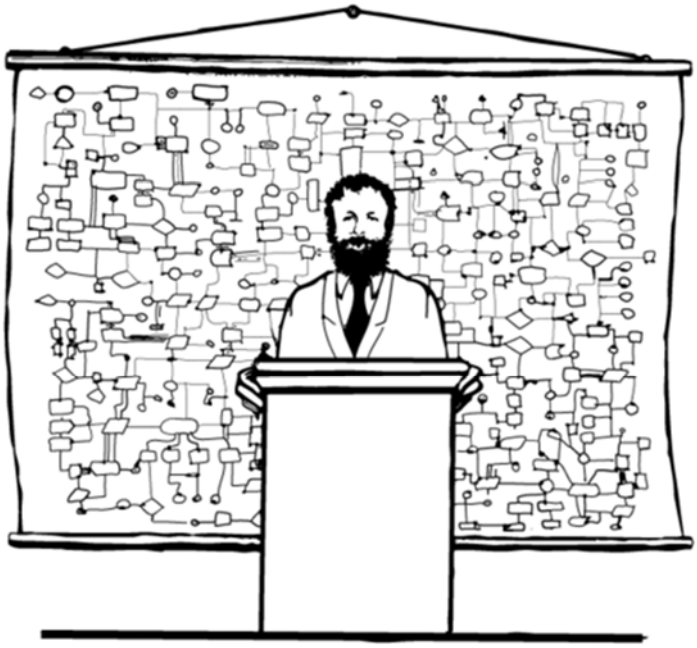


Sw Eng skills

Many kind of skills are needed for successfull software engineering...

much more than just programming or just technical skills.



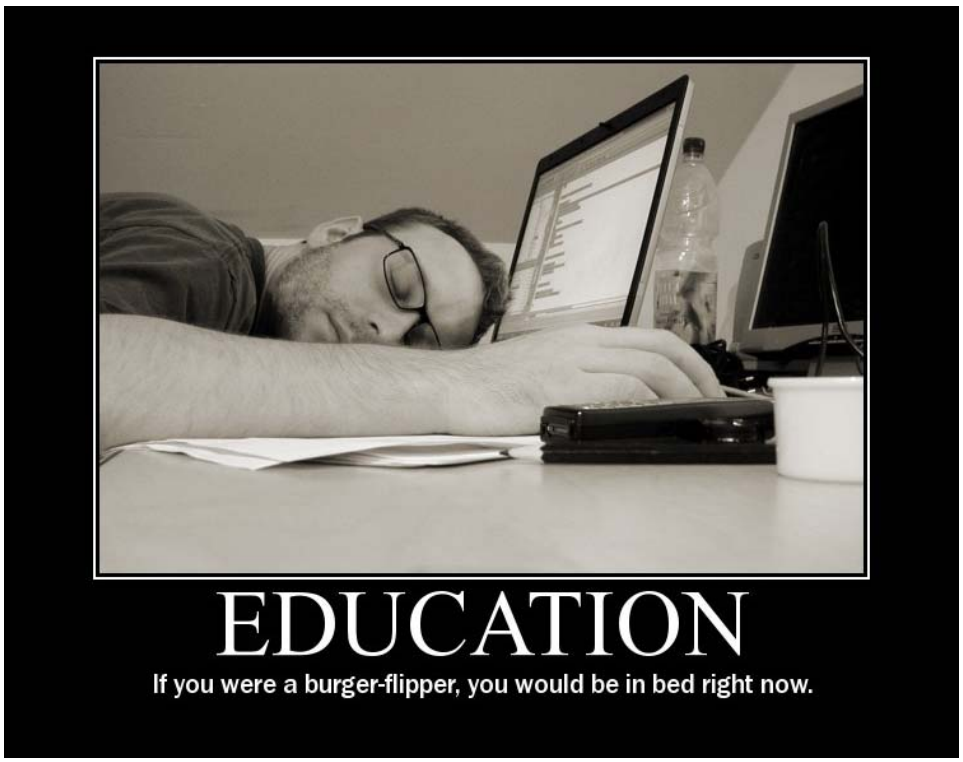


"Now that you have an overview of the system, we're ready for a little more detail"

Tampereen yliopisto
Tampere University

What to do next…?

- find a project assignment group / search more students to fullfil your group
- sign in some weekly exercise group (there may not be all five…), opens at Moodle at 1800 o'clock
-
-


EDUCATION
If you were a burger-flipper, you would be in bed right now.